ISO TC 184/SC4/ ___ Nwwww **Date:** 2000/09/29

Supersedes ISO TC 184/SC4/ __ N ___

ISO/WD 10303-5w

Product data representation and exchange: Integrated resource: Numerical analysis and support

ABSTRACT:
This provides an initial draft of a resource to support the Fluid Dynamics AP.

KEYWORDS: Mesh, Analysis

COMMENTS TO READER:
This document is very much work in progress. It is a revised version of the one issued on 2000/06/16. The formal modeling uses EXPRESS, Amendment 1.

Project Leader: Ray Cosner
Address: Boeing, Phantom Works
PO Box 516,
M/S S106–7126
St. Louis, MO 63166

Telephone: +1 (314) 233–6481
Telefacsimile: +1 (314) 777–1328
Electronic mail: raymond.r.cosner@boeing.com

Project Editor: Peter Wilson
Address: Boeing Commercial Airplane
PO Box 3707, M/S 6H-AF
Seattle, WA 98124-2207

Telephone: +1 (425) 237–3506
Telefacsimile: +1 (425) 327–3428
Electronic mail: peter.r.wilson@boeing.com

revision 7, 1/00 (PRW)

International Organization for Standardization
Case Postale 56 • CH–2111 Genève 20 • Switzerland

Contents

Page

Figures

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 3.

Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75% of the member bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this International Standard may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights.

International Standard ISO 10303–5w was prepared by Technical Committee ISO/TC 184, *Industrial automation systems and integration*, Subcommittee SC4, *Industrial data*.

This International Standard is organized as a series of parts, each published separately. The parts of ISO 10303 fall into one of the following series: description methods, integrated resources, application interpreted constructs, application protocols, abstract test suites, implementation methods, and conformance methods. The series are described in ISO 10301-1.

A complete list of parts of ISO 10303 is available from the Internet:
<http://www.nist.gov/sc4/editing/step/titles/>

This part of ISO 10303 is a member of the integrated resource series.

Annexes A and B are a normative part of this International Standard. Annexes C and D are for information only.

Introduction

ISO 10303 is an International Standard for the computer-interpretable representation and exchange of product data. The objective is to provide a neutral mechanism capable of describing product data throughout the life cycle of a product independent from any particular system. The nature of this description makes it suitable not only for neutral file exchange, but also as a basis for implementing and sharing product databases and archiving.

Major subdivisions of this International Standard are:

— **numerical_analysis_schema**;

— **multiblock_schema**.

The relationships of the schemas in this part of ISO 10303 to other schemas that define the integrated resources of this International Standard are illustrated in Figure 1 using the EXPRESS-G notation. EXPRESS-G is defined in annex D of ISO 10303-11. The schemas identified in the bold boxes are specified in this part of ISO 10303. The **product_definition_schema** and **support_resource_schema** are specified in part 41 of ISO 10303. The **representation_schema**, **mathematical_functions_schema**, and **mesh_topology_schema** are specified in parts 43, 50, and 5s of ISO 10303, respectively. The schemas illustrated in Figure 1 are components of the integrated resources.

There are many applications that, in one way or another, generate numerical solutions to engineering and mathematical problems. The applications can range from almost trivial, like solving a linear equation which, if the numbers are simple enough, can be performed mentally, to determining the characteristics of a set of experimental data approximating a normal distribution, which requires at least pencil and paper, to a computational fluid dynamics problem which may require several hours of supercomputer time to resolve.

This part of ISO 10303 provides general application independent means of representing analysis problems and the solutions of such problems.

In this International Standard the same English language words may be used to refer to an object in the real world or to a concept, and as the name of an EXPRESS data type that represents this object or concept. The following typographical convention is used to distinguish between these. If a word or phrase occurs in the same typeface as narrative text, the referent is the object or concept. If the word or phrase occurs in a bold typeface, the referent is the EXPRESS data type. Names of EXPRESS schemas also occur in a bold typeface.

The name of an EXPRESS data type may be used to refer to the data type itself, or to an instance of the data type. The disctinction between these uses is normally clear from the context. If there is a likelihood of ambiguity, the phrase 'entity data type' or 'instance(s) of' is included in the text.

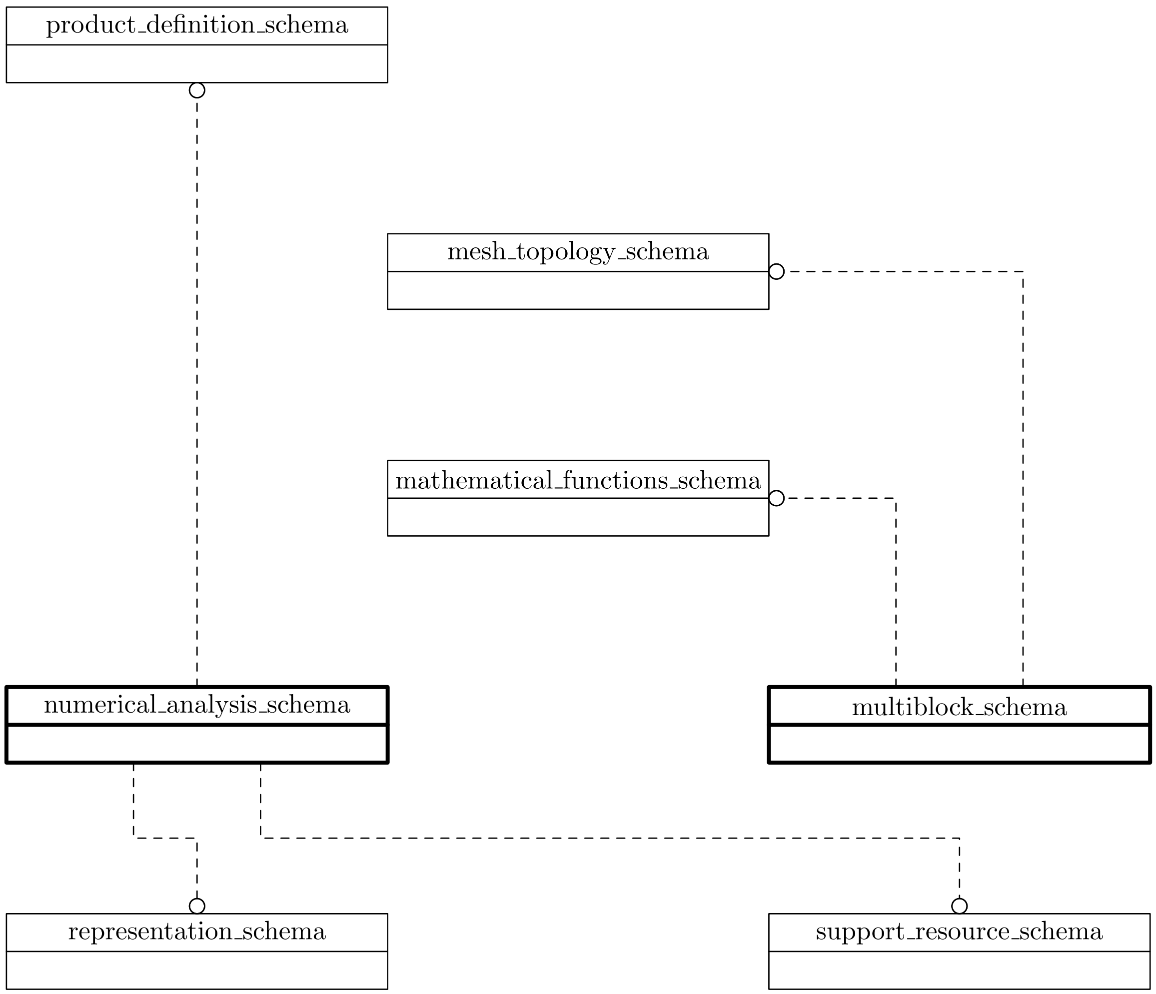

Figure 1 – Schema relationships

Quotation marks “ ” are used to denote text that is copied from another document. Inverted commas ‘’ are used to denote particular string values.

Several components of this part of ISO 10303 are available in electronic form. This access is provided through the specification of Universal Resource Locators (URLs) that identify the location of these files on the Internet. If there is difficulty accessing these sites contact the ISO Central Secretariat or the ISO TC184/SC4 Secretariat directly at: `sc4@cme.nist.gov`.

Industrial automation systems and integration — Product data representation and exchange — Part 5w : Integrated resource: Numerical analysis and support

1 Scope

The following are within the scope of this part of ISO 10303:

— Application-independent numerical analysis;

— Multiblock mesh interfaces;

The following are outside the scope of this part of ISO 10303:

— Numerical analysis applications;

— Applications of mesh interfaces.

2 Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this international standard. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this international standard are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative document referred to applies. Members of ISO and IEC maintain registers of currently valid International Standards.

ISO 10303-1:1994, *Industrial automation systems and integration — Product data representation and exchange — Part 1: Overview and fundamental principles.*

ISO 10303-11:1994, *Industrial automation systems and integration — Product data representation and exchange — Part 11: Description methods: The EXPRESS language reference manual.*

ISO/TR 10303-12:1997, *Industrial automation systems and integration — Product data representation and exchange — Part 12: Description methods: The EXPRESS-I language reference manual.*

ISO 10303-41:1994, *Industrial automation systems and integration — Product data representation and exchange — Part 41: Integrated resource: Fundamentals of product description and support.*

ISO 10303-42:1994, *Industrial automation systems and integration — Product data representation and exchange — Part 43: Integrated resource: Representation structures.*

ISO 10303-50:2000[1)], *Industrial automation systems and integration — Product data representation and exchange — Part 50: Integrated resource: Mathematical constructs.*

ISO 10303-5s:2000[1)], *Industrial automation systems and integration — Product data representation and exchange — Part 5s: Integrated resource: Mesh-based topology.*

ISO/IEC 8824-1:1995, *Information technology — Abstract Syntax Notation One (ASN.1): Specification of basic notation.*

3 Terms, definitions, abbreviations, and symbols

3.1 Terms defined in ISO 10303-1

— application protocol (AP)

3.2 Terms defined in ISO 10303-5s

— cell

— mesh

— vertex

3.3 Abbreviations

CFD	computational fluid dynamics

[1)]To be published.

4 numerical_analysis_schema

The following EXPRESS declaration begins the **numerical_anlaysis_schema** and identifies the necessary external references.

EXPRESS specification:

```
*)
{iso standard 10303 part (11) version (4)}
SCHEMA numerical_analysis_schema;
  REFERENCE FROM product_definition_schema
    (product);
  REFERENCE FROM representation_schema
    (representation_item);
  REFERENCE FROM support_resource_schema
    (label,
     text);
(*
```

NOTE The schemas referenced above can be found in the following parts of ISO 10303:

product_definition_schema	ISO 10303-41
representation_schema	ISO 10303-43
support_resource_schema	ISO 10303-41

4.1 Introduction

This schema defines and describes the structures for describing numerical analyses.

4.2 Fundamental concepts and assumptions

Numerical analyses can range from the trivial to highly complex. Nevertheless any numerical analysis can be considered to have four components:

— An analysis space within which a solution to a problem is required. For example, if the problem is to determine whether a diving springboard will break under the weight of a 400lb swimmer, there is no need to consider the overall structure of the building housing the pool, or if the problem is to simulate the flow of a product through a proposed production assembly line there is no need to consider the trucking system delivering component parts to the factory door.

— A set of one or more equations that represent the problem.

— A set of conditions that specify the intial conditions of the problem and/or constraints on the solution to the problem.

— The solution to the problem and auxiliary information relevant to the solution. Examples of potential auxiliary information include error estimates, convergence rates for problems solved iteratively, and so on.

In essence, we can consider the following entity to be representative of a numerical analysis:

```
ENTITY representative_numerical_analysis;
  domain     : domain;
  equations  : SET OF equation;
  conditions : SET OF condition;
  results    : SET OF result;
END_ENTITY;
```

Because of the very broad nature of potential numerical analyses it is not possible to define a detailed model that caters for everything. The model as specified therefore just covers, at an abstract level, the above components. It is expected that other parts of this standard such as an AP, or other standards, that make use of this part of ISO 10303 will specialise the model to meet their own requirements, by subtyping the entities and adding extra attributes and constraints as necessary.

Further, numerical analysis has a life cycle. Although at the end an analysis has all four components at the start of the life cycle none of these may be known, just the fact that there will be an analysis. The order of defining and performing a particular analysis might be:

a) specify the equations;

b) broadly decide on the conditions;

c) decide on the kind of analysis and the domain;

d) specify the conditions in terms of the domain; and

e) perform the analysis and capture the results.

Thus, depending on the analysis and the particular point in its life cycle, practically any combination of the components may be present or absent.

4.3 numerical_analysis_schema entity definitions

4.3.1 numerical_analysis

A numerical analysis may have several components, as outlined in 4.2. These are analysis space (or domain), equations, conditions, and results.

EXPRESS specification:

```
*)
```



```
ENTITY numerical_analysis
  SUBTYPE OF (representation_item);
  descriptions : OPTIONAL LIST OF text;
END_ENTITY;

SUBTYPE_CONSTRAINT sc1_numerical_analysis FOR numerical_analysis;
  ABSTRACT SUPERTYPE;
END_SUBTYPE_CONSTRAINT;
(*
```

Attribute definitions:

name: (inherited) user-specified instance identifier;

descriptions: is annotation;

4.3.2 product_numerical_analysis

A relationship between a **product** and a **numerical_analysis**.

EXPRESS specification:

```
*)
ENTITY product_numerical_analysis;
  id           : label;
  descriptions : OPTIONAL LIST OF text;
  the_product  : product;
  the_analysis : numerical_analysis;
END_ENTITY;
(*
```

Attribute definitions:

id: user-specified instance identifier;

descriptions: is annotation;

the_product: is the **product**;

the_ananysis: is the **numerical_analysis**.

4.3.3 analysis_space

The computational space for a numerical analysis.

EXAMPLE 1 Some examples are:

— The mesh for a finite element thermal analysis;

— The multiblock zones for a computational fluid dynamics analysis;

— The node points for a finite difference computation.

EXPRESS specification:

```
*)
ENTITY analysis_space;
  id           : label;
  descriptions : OPTIONAL LIST OF text;
END_ENTITY;

SUBTYPE_CONSTRAINT sc1_analysis_space FOR analysis_space;
  ABSTRACT SUPERTYPE;
END_SUBTYPE_CONSTRAINT;
(*
```

Attribute definitions:

id: user-specified instance identifier;

descriptions: is annotation.

4.3.4 analysis_equation

An equation for solution.

EXAMPLE 1 Some examples are:

— Poisson's equation;

— The diffusion equation;

— Linear constraint equation.

EXPRESS specification:

```
*)
ENTITY analysis_equation;
  id           : label;
  descriptions : OPTIONAL LIST OF text;
END_ENTITY;

SUBTYPE_CONSTRAINT sc1_analysis_equation FOR analysis_equation;
```



```
  ABSTRACT SUPERTYPE;
END_SUBTYPE_CONSTRAINT;
(*
```

Attribute definitions:

id: user-specified instance identifier;

descriptions: is annotation.

4.3.5 analysis_condition

An initial state or boundary condition.

EXAMPLE 1 Some examples are:

— Pressure loading;

— Symmetry;

— Vibration frequency.

EXPRESS specification:

```
*)
ENTITY analysis_condition;
  id           : label;
  descriptions : OPTIONAL LIST OF text;
END_ENTITY;

SUBTYPE_CONSTRAINT sc1_analysis_condition FOR analysis_condition;
  ABSTRACT SUPERTYPE;
END_SUBTYPE_CONSTRAINT;
(*
```

Attribute definitions:

id: user-specified instance identifier;

descriptions: is annotation.

4.3.6 analysis_result

A result of some computation.

EXAMPLE 1 Some examples are:

— Maximum Mach number;

— Fundamental vibration frequency;

— Stress distribution;

— Product flow rate through a production line;

— Critical path through a PERT chart.

EXPRESS specification:

```
*)
ENTITY analysis_result;
  id           : label;
  descriptions : OPTIONAL LIST OF text;
END_ENTITY;

SUBTYPE_CONSTRAINT sc1_analysis_result FOR analysis_condition;
  ABSTRACT SUPERTYPE;
END_SUBTYPE_CONSTRAINT;
(*
```

Attribute definitions:

id: user-specified instance identifier;

descriptions: is annotation.

EXPRESS specification:

```
*)
END_SCHEMA; -- end of numerical_analysis_schema
(*
```

5 multiblock_schema

The following EXPRESS declaration begins the **multiblock_schema** and identifies the necessary external references.

EXPRESS specification:

```
*)
```


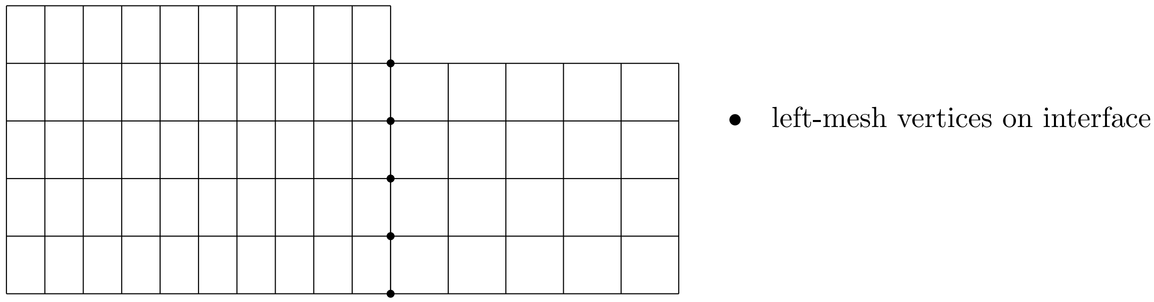

Figure 2 – A 1-to-1 abutting interface

```
{iso standard 10303 part (11) version (4)}
SCHEMA multiblock_schema;
  REFERENCE FROM data_array_schema
    (data_array,
     index_list,
     index_range);
  REFERENCE FROM mathematical_functions_schema
    (listed_real_data);
  REFERENCE FROM mesh_topology_schema
    (unstructured_mesh,
     structured_mesh);
  REFERENCE FROM support_resource_schema
    (label,
     text);
(*
```

NOTE The schemas referenced above can be found in the following parts of ISO 10303:

data_array_schema	ISO 10303-5s
mathematical_functions_schema	ISO 10303-50
mesh_topology_schema	ISO 10303-5s
support_resource_schema	ISO 10303-41

5.1 Introduction

This schema defines and describes the structures for describing multiblocks.

5.2 Fundamental concepts and assumptions

Figures 2 to Figure 4 show three types of multiblock interfaces.

Figure 2 illustrates a 1-to-1 abutting interface, also referred to as matching or C0 continuous. The interface is a plane of vertices that are physically coincident (i.e., they have identical coordinate values) between the adjacent meshes; grid-coordinate lines perpendicular to the interface are continuous from one mesh to the next. In 3-D, a 1-to-1 abutting interface is always a logically rectangular region.

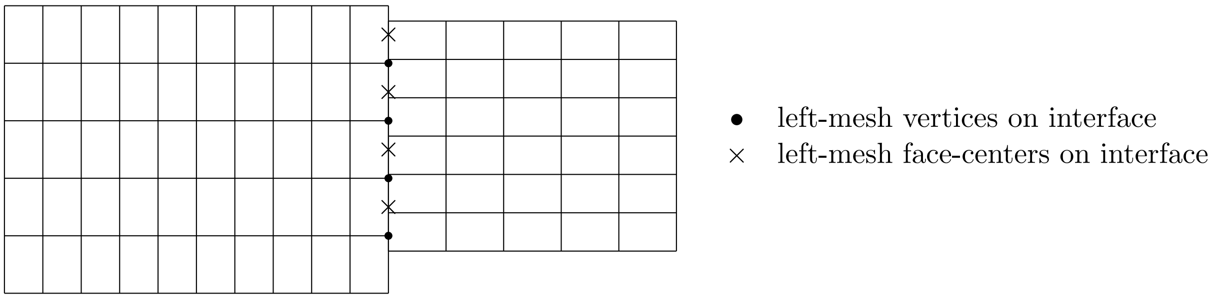

Figure 3 – A mismatched abutting interface

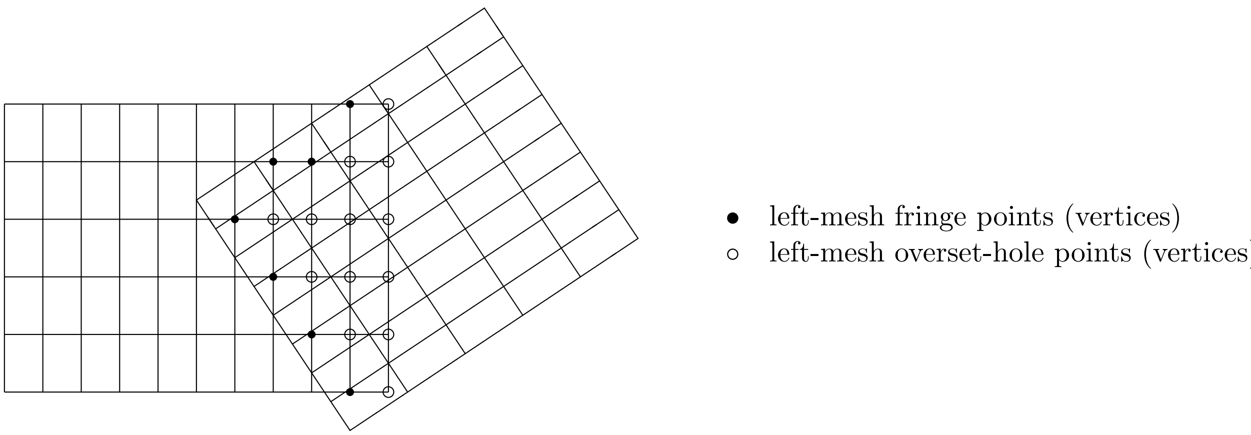

Figure 4 – An overset interface

The second type of interface, is mismatched abutting, where two meshes touch but do not overlap (except for vertices and cell faces on the grid plane of the interface). Vertices on the interface may not be physically concident between the two meshes. Figure 3 indentifies the vertices and face centers of the left mesh that lie on the interface. In 3-D, the vertices of a mesh that constitute an interface patch may not form a logically rectangular.

The third type of multiblock interface is called overset and occurs when two meshes overlap; in 3-D, the overlap is a 3-D region. For overset interfaces, one of the two meshes takes precedence over the other; this establishes which solution in the overlap region to retain and which one to discard. The region in a given mesh where the solution is discarded is called an overset hole and the grid points outlining the hole are called fringe points.

Figure 4 depicts an overlap region between two meshes, where the right mesh takes precedence over the left mesh. The points identified in Figure 4 are the fringe points and overset-hole points for the left mesh. In addition, for the mesh taking precedence, any bounding points (i.e., vertices on the bounding faces) of the mesh that lies within the overlap must also be identified.

Overset interfaces may include multiple layers of fringe points outlining holes and at mesh boundaries.

For the mismatched abutting and overset interfaces in Figure 3 and Figure 4, the left mesh plays the role of receiver mesh and the right plays the role of donor mesh.

5.3 multiblock_schema type definitions

5.3.1 grid_location

grid_location is an enumeration of locations with respect to a grid.

EXPRESS specification:

```
*)
TYPE grid_location = ENUMERATION OF
     (vertex,
      cell_center,
      face_center,
      iface_center,
      jface_center,
      kface_center,
      edge_center);
END_TYPE;
(*
```

Enumerated item definitions:

vertex: is coincident with the grid vertices;

cell_center: is the center of a cell; this is also appropriate for entities associated with cells but not necessarily with a given location in a cell;

face_center: is the center of a generic face which can point in any coordinate direction;

iface_center: is the center of a face in 3-D whose computational normal points in the i direction;

jface_center: is the center of a face in 3-D whose computational normal points in the j direction;

kface_center: is the center of a face in 3-D whose computational normal points in the k direction;

edge_center: is the center of an edge.

5.4 multiblock_schema entity definitions

5.4.1 multiblock

A multiblock is a grouping of connected meshes. All interface grid connectivity interface information pertaining to the group is contained in the **multiblock** structure. This includes abutting interfaces (general mismatched and 1-to-1), overset-grid interfaces, and overset-grid holes.

All the interface patches for a given mesh in the group are contained in the **multiblock** entity for that group. If a face of a mesh touches several other meshes (say N), the N different instances of the **block_connectivity** structure must be included in the **multiblock** to describe each interface patch.

NOTE 1 This convention requires that a single interface patch be described twice — once for each adjacent mesh. It also means that the **multiblock** is symmetrical with regard to interface patches.

EXPRESS specification:

```
*)
ENTITY multiblock;
  id             : label;
  descriptions   : OPTIONAL LIST OF text;
  connectivities : LIST OF block_connectivity;
END_ENTITY;
(*
```

Attribute definitions:

id: user-specified instance identifier;

descriptions: is annotation;

connectivities: is the connectivity information;

5.4.2 block_connectivity

Information specifying the connectivity of a multiblock interface.

EXPRESS specification:

```
*)
ENTITY block_connectivity;
  descriptions : OPTIONAL LIST OF text;
  block        : mesh;
DERIVE
```



```
  index_count  : INTEGER := block.index_count;
END_ENTITY;

SUBTYPE_CONSTRAINT sc1_block_connectivity FOR block_connectivity;
  ABSTRACT SUPERTYPE;
  ONEOF(matched_connection,
        mismatched_connection);
END_SUBTYPE_CONSTRAINT;
(*
```

Attribute definitions:

descriptions: is annotation;

block: is the current (receiver) mesh;

index_count: is the number of indices required to identify uniquely a vertex or cell in the block.

5.4.3 matched_connection

matched_connection contains connectivity information for a multiblock interface patch that is abutting with 1-to-1 matching between adjacent structured mesh indices (also referred to as C0 connectivity). An interface patch is the subrange of the face of a mesh that touches one and only one other mesh. This structure identifies the subrange of indices for the two adjacent meshes that make up the interface and gives an index transformation from one mesh to the other. It also identifies the adjacent mesh.

EXPRESS specification:

```
*)
ENTITY matched_connection
  SUBTYPE OF (block_connectivity);
  SELF\block_connectivity.block : structured_mesh;
  range       : index_range;
  donor       : structured_mesh;
  donor_range : index_range;
  transform   : ARRAY [1:index_count] OF INTEGER;
WHERE
  wr1 : block :<>: donor;
  wr2 : donor.index_count = index_count;
  wr3 : range.index_count = index_count;
  wr4 : donor_range.index_count = index_count;
END_ENTITY;
(*
```


Attribute definitions:

block: (inherited) is the current mesh;

index_count: (inherited) is the number of indices required to reference a vertex.

range: contains the subrange of indices that makes up the interface patch in the **block**;

donor: is the adjacent mesh;

donor_range: contains the interface patch subrange of indices for the **donor**;

transform: contains a shorthand notation for the transformation matrix describing the relationship between indices of the two adjacent meshes (see below).

Formal propositions:

wr1: **block** and **donor** shall be different;

wr2: The **index_count**s of **block** and **donor** shall have the same value.

wr3: The **index_count**s of **block** and **range** shall have the same value.

wr4: The **index_count**s of **donor** and **donor_range** shall have the same value.

The shorthand matrix notation used in **transform** has the following properties. The matrix itself has rank **index_count** and contains elements +1, 0, and −1; it is orthonormal and its inverse is its transpose. The transformation matrix (`T`) works as follows: If `Index1` and `Index2` are the indices of a given point on the interface, where `Index1` is in the current mesh and `Index2` is in the adjacent mesh, then their relationship is,

```
Index2 = T.(Index1 - Start1) + Start2
Index1 = Transpose[T].(Index2 - Start2) + Start1
```

where the '`.`' notation indicates matrix-vector multiply, `Start1` and `Finish1` are the subrange indices contained in **range**, and `Start2` and `Finish2` are the subrange indices contained in **donor_range**.

The short-hand notation used in **transform** is as follows. Each element shows the image in the adjacent mesh's face of a positive index increment in the current mesh's face. The first element is the image of a positive increment in i; the second element is the image of an increment in j; and the third (in 3-D) is the image of an increment in k in the current mesh's face. For 3-D, the transformation matrix T is constructed from **transform** $= [\pm a, \pm b, \pm c]$ as follows:

$$\mathtt{T} = \begin{bmatrix} \mathrm{sgn}(a)\mathrm{del}(a-1) & \mathrm{sgn}(b)\mathrm{del}(b-1) & \mathrm{sgn}(c)\mathrm{del}(c-1) \\ \mathrm{sgn}(a)\mathrm{del}(a-2) & \mathrm{sgn}(b)\mathrm{del}(b-2) & \mathrm{sgn}(c)\mathrm{del}(c-2) \\ \mathrm{sgn}(a)\mathrm{del}(a-3) & \mathrm{sgn}(b)\mathrm{del}(b-3) & \mathrm{sgn}(c)\mathrm{del}(c-3) \end{bmatrix},$$

where,

$$\mathrm{sgn}(x) \equiv \begin{cases} +1, & \text{if } x \geq 0 \\ -1, & \text{if } x < 0 \end{cases} \qquad \mathrm{del}(x-y) \equiv \begin{cases} 1, & \text{if } \mathrm{abs}(x) = \mathrm{abs}(y) \\ 0, & \text{otherwise} \end{cases}$$

EXAMPLE 1 **transform** `= [-2, +3, +1]` gives the transformation matrix,

$$\mathtt{T} = \begin{bmatrix} 0 & 0 & +1 \\ -1 & 0 & 0 \\ 0 & +1 & 0 \end{bmatrix}$$

NOTE 1 For establishing relationships between adjacent and current mesh indices lying on the interface itself, one of the elements of **transform** is superfluous since one component of both interface indices remains constant.

NOTE 2 The transform matrix and the two index pairs overspecify the interface patch. For example, `Finish2` can be obtained from **transform**, `Start1`, `Finish1` and `Start2`.

5.4.4 mismatched_connection

mismatched_connection contains conectivity information for generalized multiblock interfaces. Its purpose is to describe mismatched-abutting and overset interfaces for both structured and unstructured meshes, and can also be used for 1-to-1 abutting interfaces.

For abutting interfaces, also referred to as patched or mismatched, an interface patch is the subrange of the face of a mesh that touches one and only one other mesh. This structure identifies the subrange of indices (or array of indices) that make up the interface and gives their image in the adjacent (donor) mesh. It also identifies the adjacent mesh. If a given face of a mesh touches several (say N) adjacent meshes, then N different instances of **mismatched_connectivity** are needed to describe all the interfaces. For a single abutting interface, two instances of **mismatched_connection** are needed — one for each adjacent mesh.

For overset interfaces, this structure identifies the fringe points of a given mesh that lie in one and only one other mesh. If the fringe points of a mesh lie in several (say N) overlapping meshes, then N different instances of **mismatched_connection** are needed to describe the overlaps. It is possible with overset grids that a single fringe point may actually lie in several overlapping meshes (though in typical usage, linkage to only one of the overlapping meshes is kept). There is no restriction against a given fringe point being contained within multiple instances of **mismatched_connection**; therefore, this structure allows the description of a single fringe point lying in several overlapping meshes.

<u>EXPRESS specification:</u>

```
*)
ENTITY mismatched_connection
  SUBTYPE OF (block_connectivity);
  range    : OPTIONAL index_range;
```

```
  vertices : OPTIONAL index_list;
  gridloc  : grid_location;
WHERE
  wr1 : EXISTS(range) XOR EXISTS(vertices);
  wr2 : (NOT EXISTS(range)) XOR
        (range.index_count = index_count);
  wr3 : (NOT EXISTS(vertices)) XOR
        (vertices.index_count = index_count);
  wr4 : (EXISTS(structured_donor) AND (structured_donor.donor :<>: block)) XOR
        (EXISTS(unstructured_donor) AND (unstructured_donor.donor :<>: block));
END_ENTITY;

SUBTYPE_CONSTRAINT sc1_mismatched_connection FOR mismatched_connection;
  ABSTRACT SUPERTYPE;
  ONEOF(mismatched_region,
        overset_hole);
END_SUBTYPE_CONSTRAINT;
(*
```

Attribute definitions:

block: (inherited) is the current mesh (the receiver mesh);

index_count: (inherited) is the number of indices required to reference a vertex.

range: is the interface points within the current mesh, if and only if the set of interface points constitute a logically rectangular region.

vertices: is the interface points with the current mesh when the set of interface points does not constitute a logically rectangular region.

gridloc: is the location of indices within the current mesh described by either **range** or **vertices**. It also identifies the location of indices described by **vertices** in a donor mesh. This allows the flexibility to describe overset interfaces for cell-centered quantities.

Formal propositions:

wr1: Either **range** or **vertices** shall have values, but not both.

wr2: If **range** exists, its **index_count** shall be the same value as **index_count**.

wr3: If **vertices** exists, its **index_count** shall be the same value as **index_count**.

wr4: Either **structured_donor** or **unstructured_donor** shall have a value. The **donor** shall not be the same as the **block**.

5.4.5 mismatched_region

A mismatched connection that is abutting or overset.

EXPRESS specification:

```
*)
ENTITY mismatched_region
  SUBTYPE OF (mismatched_connection);
  structured_donor   : OPTIONAL structured_donor;
  unstructured_donor : OPTIONAL unstructured_donor;
WHERE
  wr1 : (EXISTS(structured_donor) AND (structured_donor.donor :<>: block)) XOR
        (EXISTS(unstructured_donor) AND (unstructured_donor.donor :<>: block));
END_ENTITY;

SUBTYPE_CONSTRAINT sc1_mismatched_region FOR mismatched_region;
  ABSTRACT SUPERTYPE;
  ONEOF(abutting,
        overset);
END_SUBTYPE_CONSTRAINT;
(*
```

Attribute definitions:

structured_donor: is an adjacent structured mesh.

unstructured_donor: is an adjacent unstructured mesh.

Formal propositions:

wr1: Either **structured_donor** or **unstructured_donor** shall have a value. The **donor** shall not be the same as the **block**.

5.4.6 abutting

A mismatched abutting mesh connection.

EXPRESS specification:

```
*)
ENTITY abutting
  SUBTYPE OF (mismatched_region);
END_ENTITY;
(*
```


Informal propositions:

ip1: The **range** or **vertices** shall describe a face subrange (i.e., points in a single computational grid plane);

ip2: The **structured_donor** shall also describe a face subrange;

5.4.7 overset

An overset mesh connection.

EXPRESS specification:

```
*)
ENTITY overset
  SUBTYPE OF (mismatched_region);
END_ENTITY;
(*
```

5.4.8 overset_hole

Grid connectivity for overset meshes may also include ‘holes’ within meshes, where any mesh data is ignored or ‘turned off’, because the data in some other overlapping mesh applies instead. **overset_hole** specifies those points within a given mesh that make up a hole (or holes).

EXPRESS specification:

```
*)
ENTITY overset_hole
  SUBTYPE OF (mismatched_connection);
END_ENTITY;
(*
```

NOTE 1 The grid points making up a hole within a mesh may be specified by an element in the **range** list if they constitute a logically rectangular region. Likewise further elements in the list may be used for further logically rectangular holes. The more general alternative is to use **vertices** to list all grid points making up the holes within a mesh. Using the list of **range** specifications, or using **range** in combination with **vertices**, may result in a given hole being specified more than once.

5.4.9 structured_donor

A donor mesh that is structured.

EXPRESS specification:

```
*)
ENTITY structured_donor;
  donor  : structured_mesh;
  points : listed_real_data;
  vsize  : INTEGER;
DERIVE
  index_count : INTEGER := donor.index_count;
INVERSE
  connect : mismatched_region FOR structured_donor;
END_ENTITY;
(*
```

Attribute definitions:

donor: The structured donor mesh.

points: is the image of the receiver mesh interface points in the donor mesh. These may be thought of as bi- or tri-linear interpolants (depending on **dimension**) in the computational grid of the donor mesh. FORTRAN multidimensional array ordering is used.

vsize: is the size of the data array necessary to contain the interface points;

index_count: is the number of indices required to reference a vertex.

connect: is the **mismatched_region** for which this is the **structured_donor**.

5.4.10 unstructured_donor

A donor mesh that is unstructured.

EXPRESS specification:

```
*)
ENTITY unstructured_donor;
  donor       : unstructured_mesh;
  cells       : index_list;
  interpolant : data_array;
  vsize       : index;
DERIVE
  index_count : INTEGER := donor.index_count;
  cell_dim    : INTEGER := donor.cell_dim;
INVERSE
  connect : mismatched_region FOR unstructured_donor;
END_ENTITY;
(*
```


Attribute definitions:

donor: is the unstructured donor mesh.

cells: contains the donor cell where the node is located.

interpolants: contains the interpolation factors to locate the node in the donor cell.

vsize: is the size of the data array necessary;

index_count: is the number of indices required to reference a vertex.

cell_dim: is the dimension of a cell in the mesh.

connect: is the **mismatched_region** for which this is the **unstructured_donor**.

5.5 multiblock_schema function definitions

5.5.1 index_ranges_OK

index_ranges_OK takes an aggregate of **index_range** as an argument, together with the expected value for each entry's **dimension** attribute. If the actual and expected attribute values agree then the function returns TRUE.

EXPRESS specification:

```
*)
FUNCTION index_ranges_OK(arg : AGGREGATE OF index_range;
                         num : INTEGER) : BOOLEAN;
REPEAT i := 1 TO SIZEOF(arg);
  IF (arg[i].dimension <> num) THEN
    RETURN(FALSE);
  END_IF;
END_REPEAT;
RETURN(TRUE);
END_FUNCTION;
(*
```

Argument definitions:

arg: A aggregation of **index_range**;

num: The expected value of the **dimension** attribute;

RETURNS: TRUE iff the expected and actual values match, otherwise FALSE.

EXPRESS specification:

```
*)
END_SCHEMA;  -- end of multiblock_schema
(*
```


Annex A
(normative)
Short names of entities

Table A.1 provides the short names of entities specified in this part of ISO 10303. Requirements on the use of short names are found in the implementation methods included in ISO 10303.

NOTE The short names are available from the Internet — see annex C.

Annex B
(normative)
Information object registration

B.1 Document identification

To provide for unambiguous identification of an information object in an open system, the object identifier

{ iso standard 10303 part(5w) version(-1) }

is assigned to this part of ISO 10303. The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

B.2 Schema identification

To provide for unambiguous identification of the **numerical_analysis_schema** in an open information system, the object identifier

{ iso standard 10303 part(5w) version(1) object(1) numerical-analysis-schema(1) }

is assigned to the **numerical_analysis_schema** schema (see 4). The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

To provide for unambiguous identification of the **multiblock_schema** in an open information system, the object identifier

{ iso standard 10303 part(5w) version(1) object(2) multiblock-schema(1) }

is assigned to the **multiblock_schema** schema (see 5). The meaning of this value is defined in ISO/IEC 8824-1, and is described in ISO 10303-1.

Annex C
(informative)
EXPRESS listing

This annex references a listing of the EXPRESS entity names and corresponding short names as specified in this part of ISO 10303. It also references a listing of each EXPRESS schema specified in this part of ISO 10303, without comments or other explanatory text. These listings are available in computer-interpretable form and can be found at the following URLs:

Short names: <http://www.mel.nist.gov/div826/subject/apde/snr/>
EXPRESS: <http://www.mel.nist.gov/step/parts/part5w/cd/>

If there is difficulty accessing these sites contact ISO Central Secretariat or contact the ISO TC 184/SC4 Secretariat directly at: sc4sec@cme.nist.gov.

NOTE The information provided in computer-interpretable form at the above URLs is informative. The information that is contained in the body of this part of ISO 10303 is normative.

Annex D
(informative)
EXPRESS-G diagrams

The diagrams in this annex correspond to the EXPRESS schemas specified in this part of ISO 10303. The diagrams use the EXPRESS-G graphical notation for the EXPRESS language. EXPRESS-G is defined in annex D of ISO 10303-11.

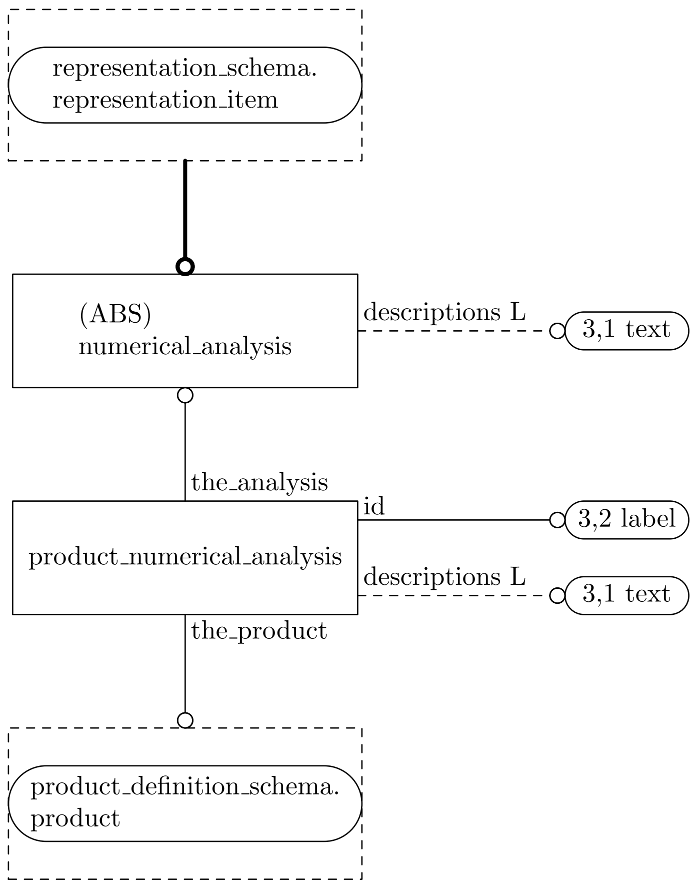

Figure D.1 – Entity level diagram of numerical_analysis_schema schema (page 1 of 3)

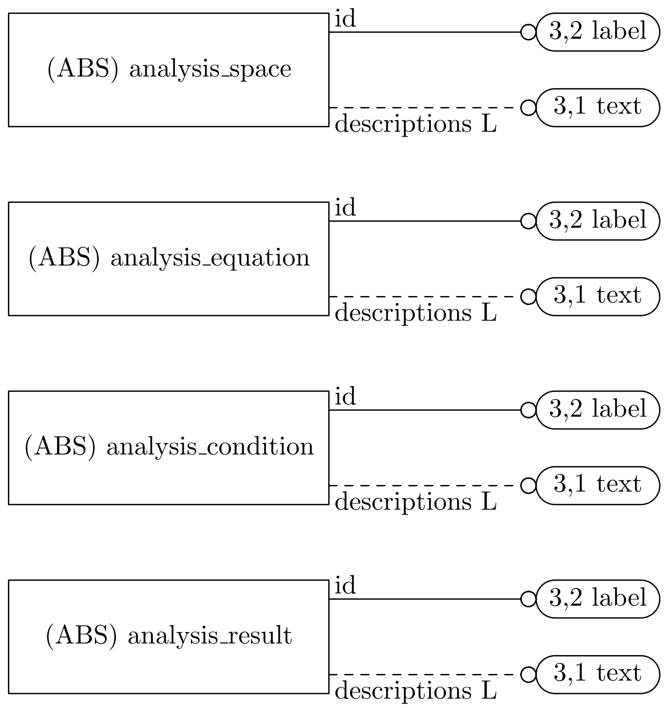

Figure D.2 – Entity level diagram of numerical_analysis_schema schema (page 2 of 3)

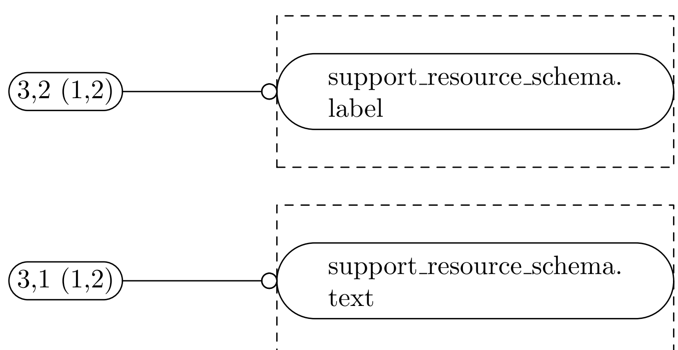

Figure D.3 – Entity level diagram of numerical_analysis_schema schema (page 3 of 3)

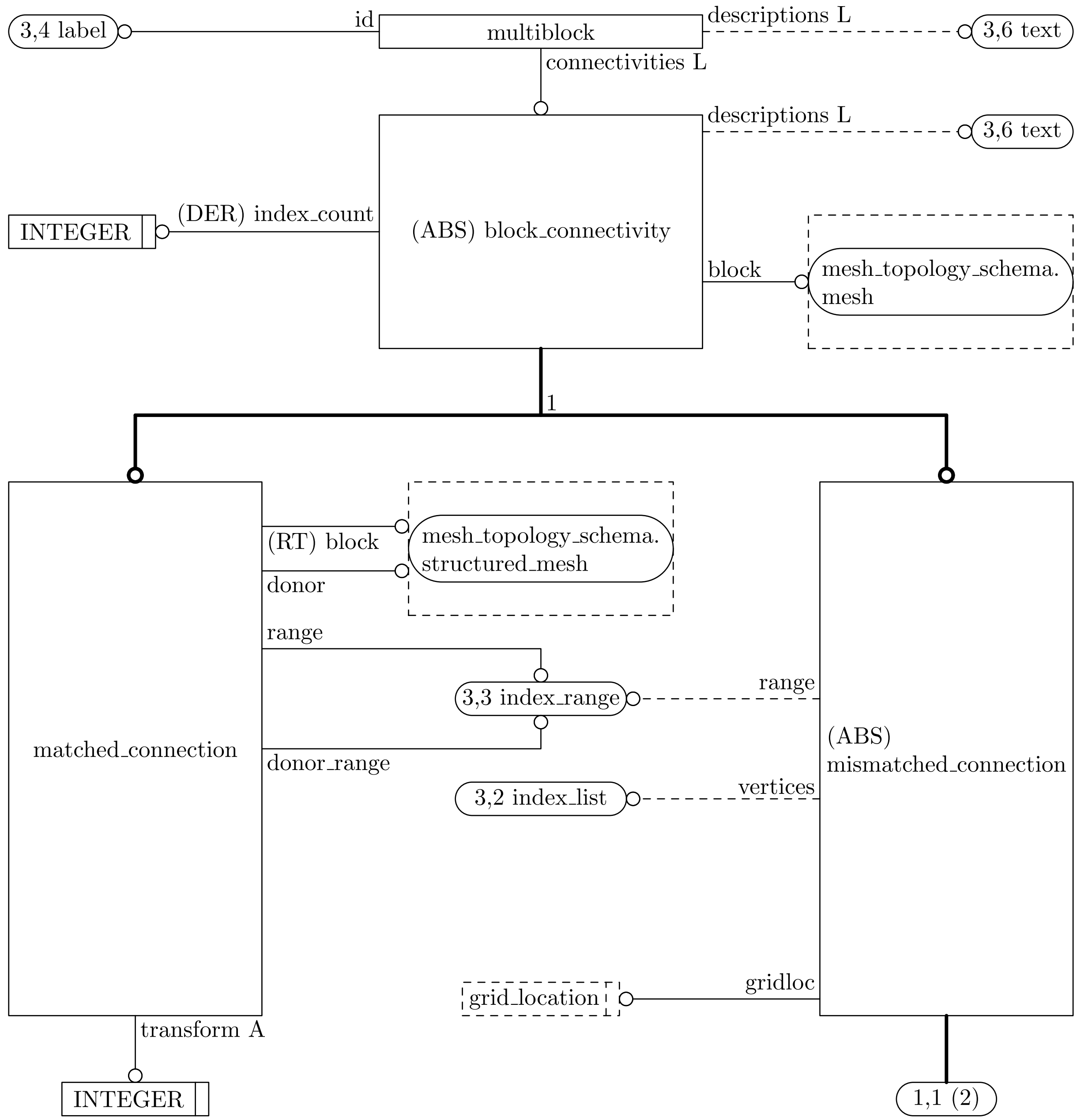

Figure D.4 – Entity level diagram of multiblock_schema schema (page 1 of 3)

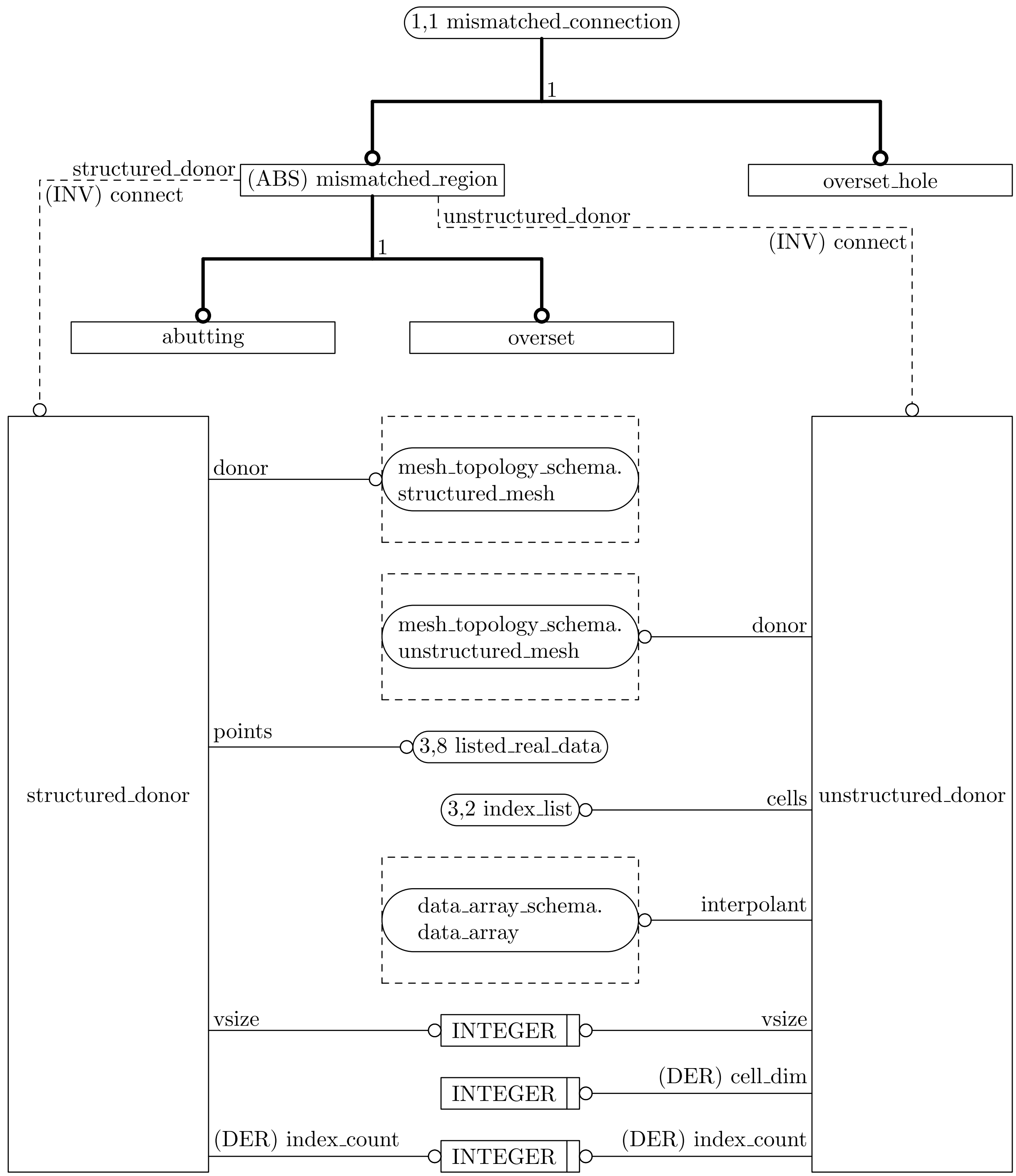

Figure D.5 – Entity level diagram of multiblock_schema schema (page 2 of 3)

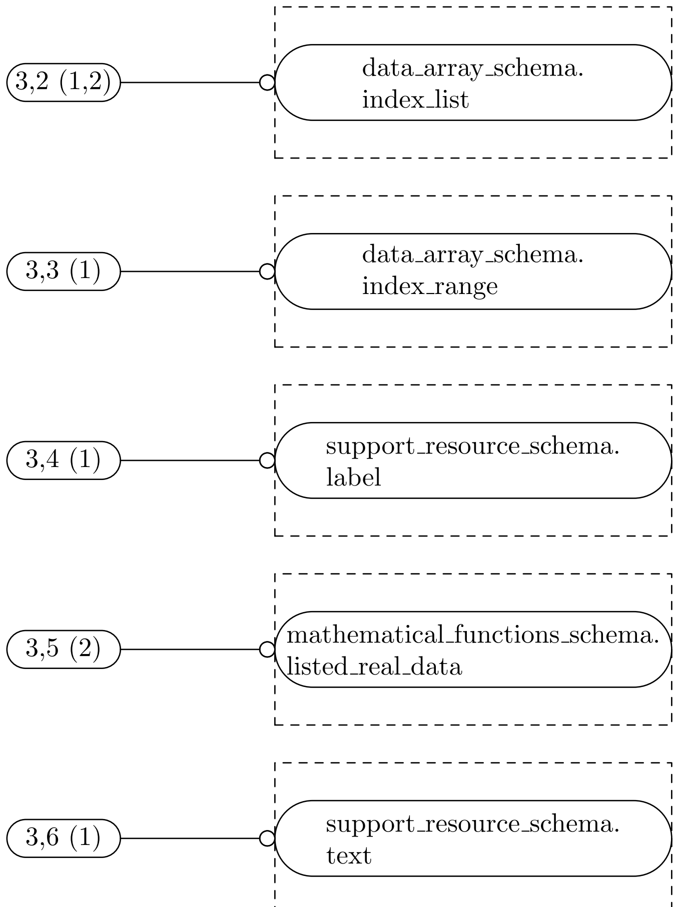

Figure D.6 – Entity level diagram of multiblock_schema schema (page 3 of 3)

Index

